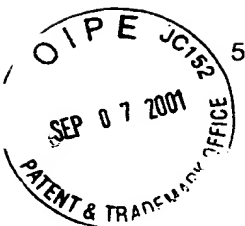# Controlled data network error recovery

[0001] This application is a Continuation of International Application PCT/FI00/00036 filed January 18, 2000 which designated the U.S. and was published under PCT Article 21(2) in English.

Field of the invention

[0002] The present invention relates to a method and equipment for transmission both in fixed networks and mobile networks, e.g. GPRS backbone networks and W-CDMA backbone networks.

Background of the invention

[0003] In packet-switched network systems a mechanism called a sliding window is used to control the flow of packets across a data link. As each packet is transmitted, an upper window edge UWE is incremented by unity. Similarly, as each packet is acknowledged, a lower window edge LWE is incremented by unity/acknowledged packet. The sending of new packets is stopped, when the difference between the UWE and LWE becomes equal to the size of the send window. Then the sending node retries to send these sent but not acknowledged packets to the same receiving node. The sending node is a packet data transmission node, which can generate packets and transfer packets other nodes have generated. If the receiving node then receives the resent packet(s) correctly, it can determine if a received packet is a valid transmitted packet or a duplicate in this simple case where only two nodes are involved. Determining is usually done by comparing the sequence number of the received packet with the sequence numbers of successfully delivered packets. Normally the sequence number is inserted into each packet by the sending node.

[0004] A problem with the current solution arises when the receiving node is "dead" for some time. In real live environments several kinds of network failures may occur or data transmitting elements may go down due to a network element failure. When that happens, the resending of packets to the same node also fails. Then the sending node reroutes the transmission and sends the packet (or packets) to another node via which it can route packets to the end system. However, it is possible that the first node received that packet (or packets) and has sent it (them) forward before the failure. The sending node does not know when the failure happened. It does not know whether the failure happened when the packet was sent first time or when the

packet was received or whether only the response (acknowledgement) got lost. Therefore, duplicates are sent to the end system. The end system has to check for every packet it receives whether it is a duplicate, for example in order not to bill a customer twice. One possible way to solve this problem is not

5   to send duplicates, but then important information may be lost.

[0005] The same problems are encountered in systems using frames, packets or any other resendable units. A frame usually comprises a protocol-related header and payload data. An empty frame is a frame with no payload data.

10   **Brief description of the invention**

[0006] The object of the invention is to overcome the problems stated above. The object of the invention is achieved with a method, a system and network nodes which are characterized by what is disclosed in the independent claims. The preferred embodiments of the invention are set forth in

15   the dependent claims.

[0007] The invention is based on indicating that a unit is possibly a duplicate of another unit which is resent because no response was received from the entity it was sent to.

[0008] The advantage of the invention is that possible duplicates

20   are indicated, so that the system can handle these units differently than other units. For example, the end system or some other system does not need to check every unit to see whether it is a duplicate. This minimizes the load in the system.

[0009] In one embodiment of the invention the sending entity is also

25   indicated. The advantage of this embodiment is that it is possible to use one and the same receiving entity as an intermediate storage for different sending entities, since the units can be identified by sending unit identity and sequence number. Therefore, they cannot be mixed up with other units having the same sequence number in the receiving entity and the network operation. A further

30   advantage is that network maintenance does not need to make sure that two sending entities do not use the same receiving entity as the intermediate storage.

[0010] In one embodiment of the invention, the unit indicated to be a possible duplicate unit is sent forward from the second entity only after the

35   sender has given instructions to send. Before giving instructions, the sender has checked from the first entity whether it has got the packet. The advantage

of this embodiment is that the duplicate unit is not sent in vain in the network and thus the network load is minimized. Another advantage of this embodiment is that there is no risk of getting duplicates of units because of a communication failure between the sending node and receiving entity. Yet another advantage of this embodiment is that the load caused by cross-checking of duplicates in a border area (e.g. a month has changed in the billing system) can be avoided.

[0011] In one embodiment of the invention, the possibility that a unit is a duplicate is checked in the end system only when the received unit has been indicated to be a possible duplicate. The advantage of this embodiment is that the duplicate checking is not done in vain and the end systems can be made more simple. Besides, the units arrive at the receiving end in such a manner that it is possible to re-establish their order.

Brief description of the figures

[0012] In the following, the invention will be described in greater detail by means of preferred embodiments and with reference to the accompanying drawings, in which

[0013] Figure 1 is a flow chart illustrating the functionality of a sending entity in the first preferred embodiment;

[0014] Figure 2 is a flow chart illustrating the functionality of a receiving entity in the first preferred embodiment;

[0015] Figure 3 is a flow chart illustrating the functionality of a sending entity in the second preferred embodiment;

[0016] Figure 4 is a flow chart illustrating the functionality of an end system according to the invention; and

[0017] Figure 5 illustrates one example of a system according to the invention.

Detailed description of the invention

[0018] The present invention is suitable for use both in fixed communications systems and in mobile communications systems. The invention is particularly suitable for use in implementing the General Packet Radio Service (GPRS) in the pan-European digital mobile communications system GSM (Global System for Mobile Communications) or corresponding mobile communications systems, such as DCS1800 and PCS (Personal Communication System). The invention is also suitable for third-generation mobile systems, such

as Universal Mobile Communication System (UMTS) and Future Public Land Mobile Telecommunication System (FPLMTS) later renamed as IMT-2000 (International Mobile Telecommunication 2000), which at present are being developed. The present invention may be used e.g. in handovers when frames

5   are also resent to the target network entity.

[0019] The present invention can be implemented in existing network nodes. They all have processors and memory with which the inventive functionality described below may be implemented. In some embodiments, some extra memory may be needed. The functions described below may be

10 located in one network element or some of them may be in one element and the others in other elements regardless of how they are located in the examples used to illustrate the invention. Transmitting nodes are also called intermediate nodes. Since the sending illustrated e.g. in figures 1 to 4 may be an internal exchange of information, nodes are also called entities. The term

15 'node' as used herein should be understood to generally refer to any network element or functionality which handles the units.

[0020] In the following description, the term 'packet' is used for the sake of clarity. The term 'packet' should be understood to also mean any other resendable unit, e.g. a frame. Frames are used e.g. in the radio link protocol.

20 In the following, the invention is described using two intermediate entities for the sake of clarity yet without limiting the invention to that kind of solutions. It is even possible to implement the invention in systems where only one receiving node exists and the sending node cannot reroute resendable units when it has enough memory. The preferable embodiments have, however, at least two al-

25 ternative directions to send these resendable units from the sending node.

[0021] In the following description, it is assumed for the sake of clarity, that a packet generating node has difficulty with a first node via which it tries to send one packet to an exit node, but it has no difficulty with a second node. The end system, e.g. a billing system, is described here to be one entity,

30 although the end system may comprise several different entities. The structure of the end system is not important to the invention. It is also assumed, for the sake of clarity, that only one packet is sent. A person skilled in the art knows how to deal with a plurality of packets being possible duplicates, that is, how to handle e.g. a window whose size is bigger than one.

35 [0022] Figure 1 illustrates the functionality of a sending node (entity) in the first preferred embodiment of the present invention. The sending node

can be a packet generating node (entity) or an intermediate node sending the packet further ahead. In step 101, the node sends a packet P1 to an entity 1. The entity 1 may be the primary peer. The packet P1 has a sequence number by which it can be identified within a window size. The window size must be smaller than the maximum sequence number in order to identify the packet properly (unless the window size is one). The node discovers in step 102 that no response is received. In step 103, it tries to resend the packet P1 to the entity 1, but fails. In another embodiment, an IPD is added to the packet P1 before resending it in step 103. The IPD is an indication indicating a possible duplicate of the packet P1. Resend means here the same as a retry send. Resend is repeated a predefined number of times after time-outs of given lengths. Then the node stores into its memory the sequence number SN of the packet P1 and the entity 1 in step 104. This is how it knows where it sent the packet P1 without receiving a response. Then, in step 105, the node picks from its priority list an entity whose priority is smaller but closest to the priority of the entity 1. Picking means that the sending node selects according to predetermined rules the next entity (or its address) to which it will try to send next. Here, that entity is the entity 2. Next, the node stores in step 106 into its memory the entity 2 so that it knows that it has first tried to send the packet P1 to the entity 1 and after that to the entity 2. Then it adds an IPD to the packet P1 in step 107. Next, the node sends in step 108 the packet P1 to the entity 2 and receives a positive response from it in step 109. A positive response means here an acknowledgement of receiving the packet P1.

[0023] If a positive response is not received from the entity 2, the node can repeat the steps 105 to 109 until a positive response is received. A positive response means that the packet P1 was received successfully. As long as the entity 1 is "dead", the packets may be sent via the entity 2.

[0024] Then in step 110 the node notices that the entity 1 is again "alive" and that packets can be sent to it again. How the node notices that the entity 1 is alive is described later in more detail with alternative examples with reference to Figure 5. In step 111, the node checks from its memory, whether there are packets sent to the entity 1, which may have a duplicate. That is, it checks e.g. whether there are packets in its buffer for unconfirmed packets. If there are no packets, the node continues normally by sending new packets either via the entity 1 or entity 2 depending of the configuration. In this example, the serving node finds out that the packet P1 is a possible duplicate and sends

a test packet to the entity 1 in step 112. This test packet is an empty packet with the sequence number of P1.

[0025] After that, the node receives a response from the entity 1 and checks in step 113 if the response is ok. If the response is ok, the entity 1
5 never got the original packet P1 or did not succeed in sending it. The response is ok, if it is e.g. a request accepted message. Therefore, there are no duplicates and the node sends in step 114 to the entity 2 a message indicating that the entity 2 can release the packet P1. In other words, the node allows the entity 2 to send the packet P1 ahead. If in step 113 the response indicates that
10 the entity 1 has already received that packet, the node sends in step 115 to the entity 2 a message indicating that the entity 2 can delete the packet P1 from its memory. A cancel message can also be used for the same purpose. In other words, the node 2 is not allowed to send the packet P1 ahead. The packet P1 is identified by the sequence number in the message sent either in
15 step 114 or 115. It is also possible to use an identification of the entity 1 in the messages sent in steps 108, 114 and 115. The message which indicates that the response is not ok may be a request already fulfilled, for example. The sending of new packets to the entity 1 is preferably done after instructions on all unconfirmed packets have been sent.

20 **[0026]** In some other embodiments, instead of the storing done in steps 104 and 105, buffering can be used, but then there is a risk of losing information due to a failure.

[0027] In some other embodiments, the whole packet P1 may be saved in step 104 and sent as a test packet in step 112. It is also possible to
25 save the packet only in the entity 2 and, when sending the test packet, the sending node first asks the entity 2 to send a copy of it. Depending on the application in these embodiments, the message sent in step 114 may only allow the release of the packets which were not sent as test packets and the test packets are deleted because their sequence numbers were not in the release
30 message.

[0028] If the resend does not fail in step 103, further packets are sent normally to the entity 1.

[0029] Figure 2 illustrates the functionality of a packet receiving entity in the first preferred embodiment. The receiving entity is assumed to be an
35 intermediate entity. The receiving entity RE receives in step 201 a packet P1 from a sending entity SE. The RE checks in step 202 whether there is an IPD

(indication of possible duplicate) in the packet P1. If there is, the RE stores in step 203 the packet P1 in order to wait for instructions from the SE. It may also store the packet P1 with the information that it received it from the SE and/or an identification of the first entity if indicated by the SE. In step 204, the RE waits for instructions. During this waiting, it may transmit other packets normally. In step 205, the RE receives instructions concerning the packet P1 from the SE. (It identifies the packet P1 e.g. by the sequence number.) In step 206, it checks if the instruction indicates a delete or a cancel. If it is a cancelling or deleting instruction, the RE deletes from its memory the packet P1 in step 207. If the instruction did not indicate a delete/cancel but a release, the RE sends the packet P1 ahead normally in step 208. In the first preferred embodiment, the packet P1 still has the IPD with the added information that it is released by an instruction from the sending entity, so that e.g. the end system may still check whether it has got it earlier, but the other intermediate nodes do not store it to wait instructions since it has this added information with the IPD. In some other embodiments, the RE may take the IPD away from the packet. Then no other checking of duplicates is done in vain. In embodiments which use maximum storing time before delivery, it is very advantageous to have the IPD in the packet, because the packet may be released because the length of the storing time of the packet reaches the maximum storing time.

[0030] If there is no IPD in step 202, the RE sends the packet P1 ahead normally in step 208.

[0031] Figure 3 illustrates the functionality of a sending node (entity) in the second preferred embodiment of the present invention. In step 301, the node sends a packet P1 to an entity 1. The node discovers in step 302 that no response has been received. Then, in step 303 the node picks from its address list an address of the next entity, entity 2. Then it adds an IPD (an indication indicating possible duplicate of the packet P1) to the packet P1 in step 304 and sends in step 305 the packet P1 to the entity 2. In some other embodiments, step 303 may be similar to step 105 of Figure 1 and/or step 103 of Figure 1 may be done between steps 302 and 303. It is assumed that after step 305 a positive acknowledgement (response) is received. If not, then the steps 303 to 305 are repeated until a positive response is received.

[0032] The packet P1 is sent ahead to the intermediate entities (nodes) in the second preferred embodiment until it reaches the end system. Figure 4 illustrates the functionality of an end system in the second preferred

embodiment of the invention. In embodiments having only one receiving node, the functionality described in Figure 4 may be implemented into it. When an embodiment related to the first embodiment of the invention is used, where the receiving entity does not remove the IPD when sending a packet forward, the
5   end system does not need to function as illustrated here in Figure 4.

[0033] Referring to Figure 4, the end system ES receives a packet P1 in step 401 and checks in step 402 whether there is an IPD (indication of possible duplicate) in the packet P1. If there is, it goes in step 403 through all the packets it has received in order to find out whether it has already received
10   this packet P1. If the ES finds out in step 404 that the packet P1 is a duplicate, it deletes in step 405 the packet P1 with the IPD it received in step 401. If the ES finds out in step 404 that it has not received the packet P1, in other words, the packet P1 is not a duplicate, it saves in step 406 the packet P1 or at least enough information in order to do a duplicate check, if needed. Then it sends
15   the packet P1 or its information for further processing according to normal procedures depending on the application. If in step 402 no IPD is found, the ES continues from step 406.

[0034] The first preferred embodiment is very well suited for applications where the order of packets in the end system is not important, like bill-
20   ing systems or email. Its advantage is that the network is not loaded unnecessarily by sending duplicates through the whole network. The second preferred embodiment may also be used in systems where the order of the packets is of some importance, like in connection with still pictures or photos.

[0035] The steps have not been set out in absolute time sequence
25   in Figures 1, 2, 3 and 4. Some of the steps described above may take place simultaneously or in a different order or some of the steps can be skipped over, e.g. steps 110 to 115. It is also possible to add new steps not shown in the figures, for example in Figure 1 between steps 109 and 110 new packets can normally be sent to the entity 2 without marking them as duplicates. It is
30   also possible to check before step 203 in Figure 2 whether the IPD includes additional information that the packet is released by an instruction from the sending entity, and if there is, the process continues from step 208, otherwise from step 203. Another possibility is to combine steps in the figures when making a new embodiment. For example, it is possible to further process the
35   packet in step 406 by taking the IPD away and sending the packet ahead. It is essential that the possibility of the packet being a duplicate is indicated. The

indication can be done e.g. by adding it to the packet header or to the payload or to the file name when file protocols are used or by sending a message indicating that the following packet is a possible duplicate. The indication may also be in another frame. It is also possible to indicate the duplicate in the message the unit is sent with, e.g. 'send packet' means that the packet is not a duplicate whereas 'send possibly duplicated packet' indicates a possible duplicate. The indication may even go via another link. It is not important how this indication is done, the essential thing is that it is done. The messages may include more information than what is stated above. The names of the messages may differ from those set out above or the indications or the instructions according to the invention may be sent in other messages than those stated above. For example, 'delete' may be 'cancel' or the IPD may be called a mark of a potential/possible duplicate MPD.

[0036] In the above, storing means that the information is stored so that it is not lost e.g. during a restart. In other words, it is stored to a nonvolatile memory. The information may be stored in the sending unit and/or any other active entity with which the sending unit has a connection. That entity may be the receiving entity or a totally different entity. In the above, a sequence number is used to identify the packet. Other identification may also be used. In a preferred embodiment, the sending unit also indicates the first receiving entity when it indicates a possible duplicate. This way, the receiving entity knows whose possible duplicates it has. This is very advantageous since the same intermediate entity can store possible duplicates first sent to different nodes and yet identify them properly. It is possible to indicate the sending node and use this information for identifying purposes.

[0037] Figure 5 illustrates one example of a system according to the invention. For the sake of clarity, Figure 5 has only one packet generating node PGN1, although it is possible to have a plurality of PGNs. The PGN1 has, in this illustrative example, three links: Link 1 to packet receiving node PRN1, Link 2 to packet receiving node PRN2 and Link 3 to packet receiving node PRN3. The PGN1 can send packets to the end system ES via all the packet receiving nodes. The packet receiving nodes are intermediate entities. The packets are sent ahead until they reach the end system ES. Although not illustrated in Figure 5, there may be any number of PRNs between e.g. PRN1 and ES. If the system illustrated in Figure 5 is a GPRS billing system, then the PGN1 may be a serving GPRS support node SGSN or a GPRS gateway sup-

port node GGSN, and packet receiving nodes may be different nodes which have a charging gateway functionality CGF. So they may also be called charging gateway nodes. The end system ES may be a billing system. The GPRS billing system with one charging gateway is described in more detail in Finnish

5   Patent 102232. This patent is incorporated herein by reference.

[0038] Some alternative examples are described below in more detail with reference to Figure 5. The abbreviations used are:

[0039] BS = Billing System

[0040] CDMA = Code Division Multiple Access

10   [0041] CDR = Call Detail Record

[0042] CGF = Charging Gateway Function

[0043] IMSI = International Mobile Subscriber Identity

[0044] GPRS = General Packet Radio Service

[0045] NE = Network Element

15   [0046] O&M = Operations and Maintenance

[0047] PDP = Packet Data Protocol

[0048] W-CDMA = Wideband CDMA

[0049] PGN = Packet Generating Node

[0050] PRN = Packet Receiving Node

20   [0051] ES = End System

[0052] The application areas in the following examples are environments where it is not absolutely crucial that the packet sent from the PGNs to the ES arrives in exactly the original order, but where it is useful that the packet contents are not lost even in abnormal network link failure situations or

25   NE failure situations. For example, charging data collection in packet-data based telecommunications systems is a very likely application area. One example of this kind of an environment is GPRS charging. In GPRS, the SGSNs and GGSNs are PGNs, the CGFs are the PRNs and the BS is the ES. Each packet transmitted between a PGN and PRN may contain one or more CDRs

30   as payload inside the packet frame.

[0053] Figure 5 is an architectural example of a chain of network elements, where the PGN1 sends packets towards the ES via either the PRN1, PRN2 or PRN3. The PRN1 is here assumed to be the primary choice (priority 1 PRN peer name configured to its PRN address list as the first place

35   to attempt packet sending). The packet flow is assumed to be as follows: Packet Generating Node(s) -> Packet Receiving Nodes -> End System.

[0054] Both the packet receiving node and the end system have a mass memory for packets.

[0055] The initial assumptions are as follows:

[0056] - The topmost protocol in the communication software stack that transfers packets between the PGN and the PRNs is assumed to be a Request-Response type message-based protocol.

[0057] - The packet send window size per each link from the packet generating node is smaller than the maximum sequence number (that is allowed to run over back to 0 and increase again per each packet).

[0058] - In most telecommunication systems, such as those given in these examples, the mass storage devices can be assumed to maintain their information even when the network element goes down due to a software failure or lack of processing capacity in relation to the traffic load.

[0059] Possible problems which are solved here in these examples are as follows:

[0060] - Duplicated information (e.g. packets containing CDRs) may be generated when traffic is redirected and the packet generating node does not surely know if the redirected packets were already successfully transmitted to the packet receiving node 1 or not.

[0061] - A network operator is governed by the laws and administration of the country within which it operates. The operators are audited by officials. An operator might be subjected to penalties or even lose its network operator licence if it generated too big bills to its subscribers, because of duplicated CDR information.

[0062] - Also, it could lose its credibility among its customers if some user data packets or CDRs related to them were either duplicated or lost.

[0063] - On the other hand, operators do not want to unnecessarily lose money, so they do not want to unnecessarily cancel packets containing CDRs unless they are 100% sure the packets are duplicates.

Example A

[0064] In this example, sequence numbers are used for packets (typically in the frame of the packets) in each link from the packet generating node (PGN) to the packet receiving node (PRN). The sequence numbers are incremented by one and roll over again to 0 after e.g. 65535, but the important

thing is that the maximum sequence number is bigger than the maximum receive window size of a PRN.

[0065] The packets are redirected (marked with a potential duplicate flag) to a parallel node PRN2 (or PRN3 if PRN2 is not available for some reason, etc.) in case the PRN1 fails. The PGN1 also sends to the PRN2 identification information of the PGN1, so that the PRN2 can identify these packets. This is necessary since the PRN2 can have packets marked with a potential duplicate flag also from other PGNs with the same sequence numbers.

[0066] The PRNs keep the packets marked potentially duplicate in memory buffers or a mass storage so that the information of their origin (PGN1 or PGN2 or other PGN) can also be associated with the packets. This is necessary since the sequence numbers of packets are unique at a certain moment only in each PGN-PRN communication link, but not necessarily at the same time in the whole network.

[0067] Packet generating nodes keep track per each packet receiving node, i.e. per each link, of sequence numbers of packets whose successful transmission is not certain. If the PGN1 fails to send a packet to the PRN1 and also fails in sending a possible duplicate of the packet to the PRN2, then the PGN1 tries to send the possible duplicate of the packet to another PRN, e.g. PRN3. However, there is no need to keep track of sequence numbers of the possible duplicates which the PGN tried to send to PRN2 per this link, since link-specific information is maintained on these packets in the link to the PRN1. It is, however, possible to add to the information relating to the link PRN1 that these packets were also sent to the PRN2 and PRN3.

[0068] The PGNs sense when their peer nodes, PRNs, come alive again. Either the PGNs send 'keep alive' messages to the PRNs at appropriate intervals (getting echoing response back from the PRNs if the PRNs are alive and working ok), or the PRNs (and possibly other nodes too) can inform their peer nodes (configured communication partners) always when they come alive after being non-working. Also, when a node is stopping working, e.g. when the operator wants to stop it to make a software update, the node can inform its peer nodes that it is going down and should not receive packets any more until it announces it has become alive again.

[0069] The PGNs sense what information the previously collapsed node had been able to process successfully.

13

[0070] The PGNs inform the secondary receiving node which packets it can send forward towards the end system.

[0071] It might also occur that at some time the PGN node goes down and its volatile memory contents are destroyed, including the buffer for
5   sequence numbers of packets whose reception by e.g. a PRN1 was possibly not successful and which have been sent (marked as potential duplicates) to a PRN2 to wait for a later decision by the PGN. In this case, the potential duplicates might stay very long at the PRN2. Therefore, it is recommended that the PRNs either have a long enough time-out to cancel such packets themselves
10   or the operator has tools for getting information about this kind of a situation and a tool to delete such long-waiting potential duplicates by an O&M operation from a PRN2. Another alternative is to finally allow the sending of the packets (e.g. marked as potential duplicates) towards the ES.

Example B

15   **[0072]** Similar as example A but there is a different sensing method for finding out whether the PRN1 (who had gone down and then become alive again) has already successfully handled a packet (sent by the PGN1) with a certain a sequence number. Here, the PGN1 would send a normal whole packet again to the PRN1, the only difference being that now the packet is
20   marked a potential duplicate. This requires that the intermediate storage for whole packets resides either in the PGN1, or the whole (potentially duplicated) packet should first be fetched from the PRN2 (where it has been waiting in an intermediate storage for a final decision whether it can be sent to the ES). The PGN1 could ask the PRN2 to give a specific potentially duplicated packet back
25   by referring to the sequence number of the packet, and after getting back the potentially duplicated packet from the PRN2, it could be resent to the PRN1. After that the PRN1 would give a response to the PGN1 (on whether it has already done the sending, i.e. processed successfully that packet or whether it got the packet "for the first time"). Then there are two possible submecha-
30   nisms in case it was "new" to the PRN1: B1) the PRN1 processes such a packet towards the ES and informs the PGN1 and the PGN1 cancels the potentially duplicated packet stored as a backup in the PRN2 (or even the PGN1), or B2) the PRN1 keeps potentially duplicated packets in its intermediate memory until the PGN1 makes the selecting commands (to cancel the
35   packet in the PRN1 and release towards the ES the potentially duplicated packet stored in the PRN2, or vice versa).

Advantages of the examples

[0073] One advantage is an improved performance in the receiving end system, since either it has to do a check for duplicated packets (containing e.g. CDRs) for only the very small minority of packets that have been produced in an abnormal case of a network node or link failure and marked as potential duplicates or no packets produced are duplicated in the PGN-PRN interface even in abnormal node or link failure events.

[0074] Another advantage is that reliability increases as regards the information contents received.

[0075] Yet another advantage is the reduction of possible manual-error recovery procedures.

[0076] The examples presented here do not require that the receive window sizes of the PRN in a network are the same, so the O&M events that configure the window size are easier to produce in practice, since all the receive window sizes of the PRNs need not be updated simultaneously.

[0077] It is possible to use as an intermediate storage for different PGNs one and the same PRN, since the packets are identified at least with the PGN and sequence number. Therefore, they are not mixed with other packets in the PRN and the O&M does not need to make sure that two PGNs do not use the same PRN as the intermediate storage.

Important Features described in the examples are:

[0078] 1) The two sensing methods presented here in examples A and B that the PGN can use to find out whether the PRN has successfully received and processed the packets that it received before the Link 1 or the PRN became malfunctioning.

[0079] 2) The buffering of the sequence numbers of packets sent from the PGN to the PRN1 associated with the not-successfully-confirmed packets sent by the PGN1. This buffer is maintained in each PGN for each PRN that the PGN is allowed to be connected to.

[0080] 3) The buffering of the sequence numbers of possibly duplicated packets sent to a PRN2 to wait for a later decision (Cancel or Release) by the PGN1.

[0081] 4) The similar redundancy method, but with the difference that the buffers for potential duplicates for each PGN1 -> PRNx reside in the PGN1 itself.

[0082] 5) The idea of marking (e.g. to packet frames) the potential duplicates, allowing them to be handled differently than the other packets (e.g. to wait in some node for final confirmation on whether they are allowed to be sent to the ES or whether the packets should be cancelled, i.e. deleted). This is the only essential feature for this invention.

[0083] 6) The idea of identifying the packets in the intermediate PRN by using the identification of the PGN and a sequence number.

[0084] The accompanying drawings and the description pertaining to them are only intended to illustrate the present invention. Different variations and modifications to the invention will be apparent to those skilled in the art, without departing from the scope and spirit of the invention defined in the appended claims.